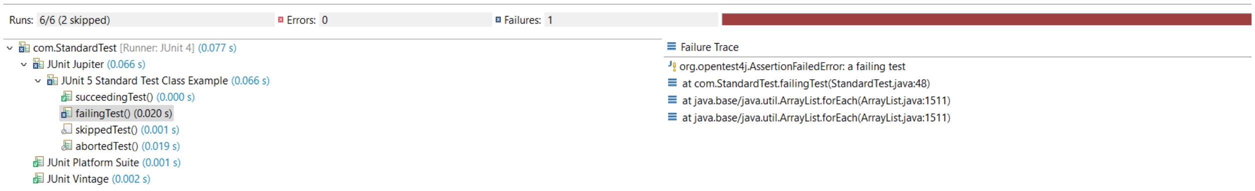
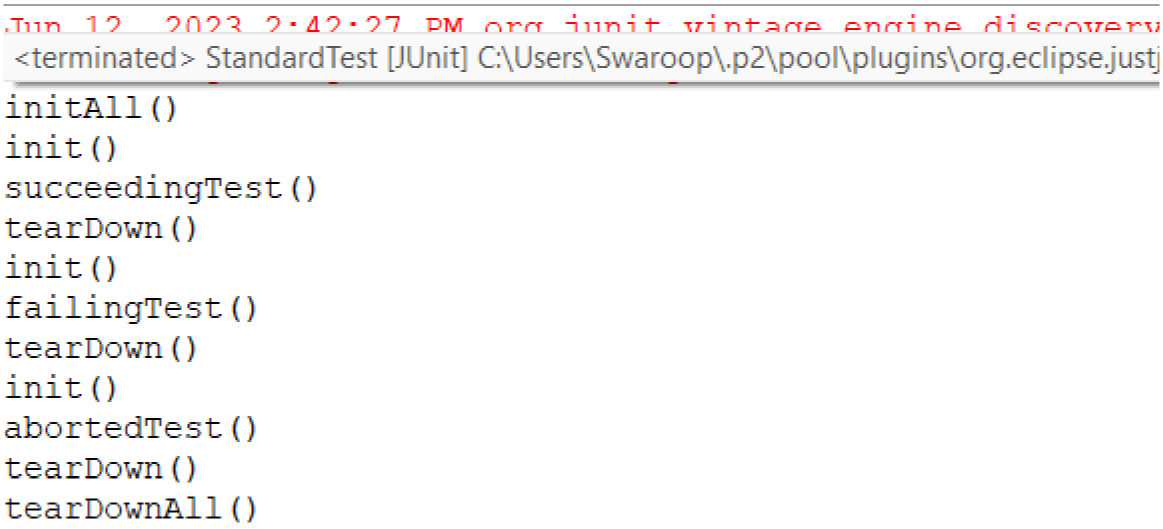
Problem Statement-1

```
package com;
import static org.junit.jupiter.api.Assertions.fail;
import static org.junit.jupiter.api.Assumptions.assumeTrue;
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Disabled;
import org.junit.jupiter.api.DisplayName;
import org.junit.jupiter.api.Test;
import org.junit.runner.RunWith;
import org.junit.platform.runner.~~JUnitPlatform~~;
@DisplayName("JUnit 5 Standard Test Class Example")
@RunWith(~~JUnitPlatform~~.class)
public class StandardTest {
@BeforeAll
static void initAll() {
System.out.println("initAll()");
}
@BeforeEach
void init() {
System.out.println("init()");
}
@Test
void succeedingTest() {
System.out.println("succeedingTest()");
}
@Test
void failingTest() {
System.out.println("failingTest()");
fail("a failing test");
}
@Test
@Disabled("for demonstration purposes")
void skippedTest() {
// not executed
}
@Test
void abortedTest() {
System.out.println("abortedTest()");
assumeTrue("abc".contains("Z"));
fail("test should have been aborted");
}
@AfterEach
void tearDown() {
System.out.println("tearDown()");
}
```

```java
@AfterAll
static void tearDownAll() {
System.out.println("tearDownAll()");
}
}
```



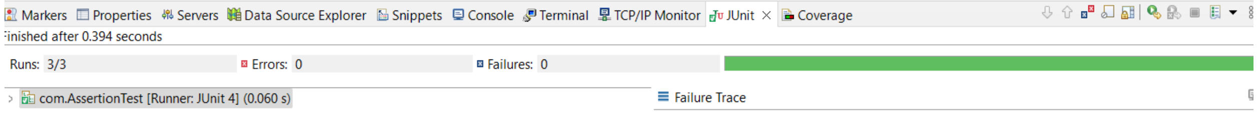
```
Runs: 6/6 (2 skipped)    Errors: 0    Failures: 1

com.StandardTest [Runner: JUnit 4] (0.077 s)
  JUnit Jupiter (0.066 s)
    JUnit 5 Standard Test Class Example (0.066 s)
      succeedingTest() (0.000 s)
      failingTest() (0.020 s)
      skippedTest() (0.001 s)
      abortedTest() (0.019 s)
  JUnit Platform Suite (0.001 s)
  JUnit Vintage (0.002 s)

Failure Trace
org.opentest4j.AssertionFailedError: a failing test
at com.StandardTest.failingTest(StandardTest.java:48)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511)
```



```
Jun 12, 2023 2:42:27 PM org.junit.vintage.engine.discovery
<terminated> StandardTest [JUnit] C:\Users\Swaroop\.p2\pool\plugins\org.eclipse.justj
initAll()
init()
succeedingTest()
tearDown()
init()
failingTest()
tearDown()
init()
abortedTest()
tearDown()
tearDownAll()
```

Problem Statement-2

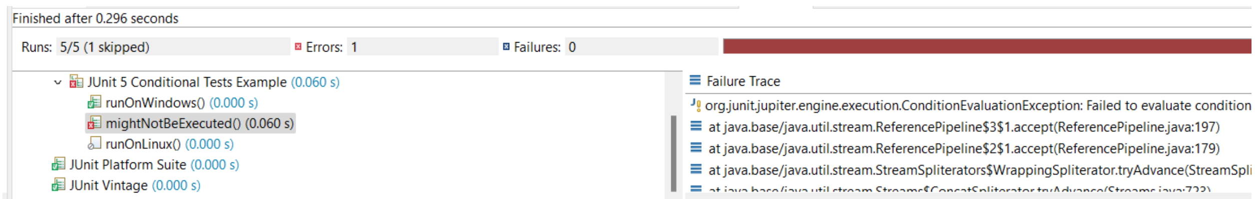
```java
package com;
import org.junit.jupiter.api.*;
import org.junit.jupiter.api.condition.DisabledIf;
import org.junit.jupiter.api.condition.EnabledOnOs;
import org.junit.jupiter.api.condition.OS;
import org.junit.platform.runner.~~JUnitPlatform~~;
import org.junit.runner.RunWith;
import static org.junit.jupiter.api.Assertions.assertArrayEquals;
import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertFalse;
import static org.junit.jupiter.api.Assertions.assertNotEquals;
import static org.junit.jupiter.api.Assertions.assertNotNull;
import static org.junit.jupiter.api.Assertions.assertNotSame;
import static org.junit.jupiter.api.Assertions.assertNull;
import static org.junit.jupiter.api.Assertions.assertSame;
import static org.junit.jupiter.api.Assertions.assertThrows;
import static org.junit.jupiter.api.Assertions.assertTrue;
@DisplayName("JUnit 5 Assertions Example")
@RunWith(~~JUnitPlatform~~.class)
public class AssertionTest {
@Test
@DisplayName("assert Examples")
public void assertTests() {
String str = null;
String str2 = "some value";
String[] a1 = { "A", "B" };
String[] a2 = { "A", "B" };
assertTrue(4 > 0);
assertFalse(5 < 1);
assertNull(str);
assertNotNull(str2);
assertSame(str, str);
assertNotSame(str, str2);
assertEquals(5, 5);
assertNotEquals(5, 6);
assertArrayEquals(a1, a2);
}
}
```



```
Finished after 0.394 seconds
Runs: 3/3    Errors: 0    Failures: 0
> com.AssertionTest [Runner: JUnit 4] (0.060 s)
```

Problem Statement-3

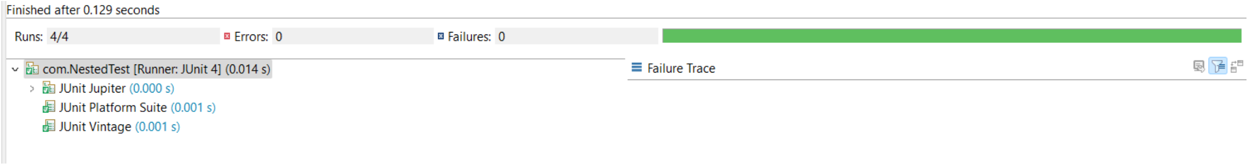
```java
package com;
import org.junit.jupiter.api.*;
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.condition.DisabledIf;
import org.junit.jupiter.api.condition.EnabledOnOs;
import org.junit.jupiter.api.condition.OS;
import org.junit.platform.runner.~~JUnitPlatform~~;
import org.junit.runner.RunWith;
@DisplayName("JUnit 5 Conditional Tests Example")
@RunWith(~~JUnitPlatform~~.class)
public class ConditionalTest {
@Test
@EnabledOnOs({OS.WINDOWS})
public void runOnWindows() {
System.out.println("This runs on Windows");
}
@Test
@EnabledOnOs({OS.LINUX})
public void runOnLinux() {
System.out.println("This runs on Linux");
}
@Test
@DisabledIf("Math.random() < 0.314159")
void mightNotBeExecuted() {
System.out.println("This may or not execute ");
}
}
```



```
Finished after 0.296 seconds
Runs: 5/5 (1 skipped)     Errors: 1     Failures: 0

JUnit 5 Conditional Tests Example (0.060 s)
    runOnWindows() (0.000 s)
    mightNotBeExecuted() (0.060 s)
    runOnLinux() (0.000 s)
JUnit Platform Suite (0.000 s)
JUnit Vintage (0.000 s)

Failure Trace
org.junit.jupiter.engine.execution.ConditionEvaluationException: Failed to evaluate condition
at java.base/java.util.stream.ReferencePipeline$3$1.accept(ReferencePipeline.java:197)
at java.base/java.util.stream.ReferencePipeline$2$1.accept(ReferencePipeline.java:179)
at java.base/java.util.stream.StreamSpliterators$WrappingSpliterator.tryAdvance(StreamSpli
at java.base/java.util.stream.Streams$ConcatSpliterator.tryAdvance(Streams.java:723)
```

Problem Statement-4

```
package com;
import org.junit.jupiter.api.*;
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.Test;
import org.junit.platform.runner.~~JUnitPlatform~~;
import org.junit.runner.RunWith;
@DisplayName("JUnit 5 Nested Example")
@RunWith(~~JUnitPlatform~~.class)
public class NestedTest {
@BeforeAll
static void beforeAll() {
System.out.println("Before all test methods");
}
@BeforeEach
void beforeEach() {
System.out.println("Before each test method");
}
@AfterEach
void afterEach() {
System.out.println("After each test method");
}
@AfterAll
static void afterAll() {
System.out.println("After all test methods");
}
@Nested
@DisplayName("Tests for the method A")
class A {
@BeforeEach
void beforeEach() {
System.out.println("Before each test method of the A class");
}
@AfterEach
void afterEach() {
System.out.println("After each test method of the A class");
}
@Test
@DisplayName("Example test for method A")
void sampleTestForMethodA() {
System.out.println("Example test for method A");
}
@Nested
@DisplayName("When X is true")
class WhenX {
@BeforeEach
void beforeEach() {
System.out.println("Before each test method of the WhenX class");
}
@AfterEach
void afterEach() {
System.out.println("After each test method of the WhenX class");
}
@Test
@DisplayName("Example test for method A when X is true")
void sampleTestForMethodAWhenX() {
System.out.println("Example test for method A when X is true");
}
}
}
}
```



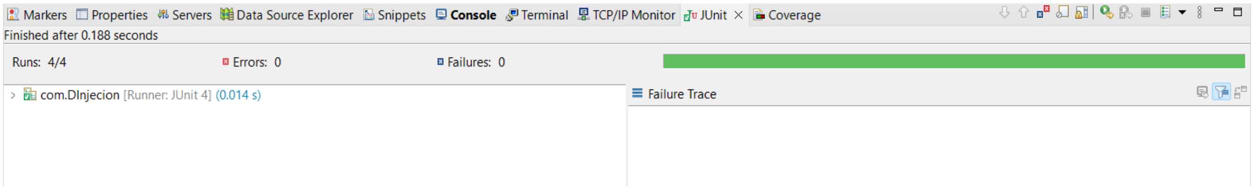
```
Finished after 0.129 seconds
Runs: 4/4    Errors: 0    Failures: 0

com.NestedTest [Runner: JUnit 4] (0.014 s)
  JUnit Jupiter (0.000 s)
  JUnit Platform Suite (0.001 s)
  JUnit Vintage (0.001 s)
```

Problem Statement-5

```
package com;
import java.util.Arrays;
import java.util.Collection;
import java.util.List;
import org.junit.jupiter.api.*;
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.condition.DisabledIf;
import org.junit.jupiter.api.condition.EnabledOnOs;
import org.junit.jupiter.api.condition.OS;
import org.junit.jupiter.api.function.Executable;
import org.junit.platform.runner.~~JUnitPlatform~~;
import org.junit.runner.RunWith;
import org.junit.jupiter.api.DynamicTest.*;
import org.junit.jupiter.api.TestFactory;
@DisplayName("JUnit 5 Dynamic Tests Example")
@RunWith(~~JUnitPlatform~~.class)
public class DynamicTest {
@TestFactory
Collection<DynamicTest> dynamicTests() {
return Arrays.asList(
dynamicTest("simple dynamic test", () -> assertTrue(true)),
dynamicTest("My Executable Class", new MyExecutable()),
dynamicTest("Exception Executable", () -> {throw new Exception("Exception
Example");}),
dynamicTest("simple dynamic test-2", () -> assertTrue(true))
);
}
}
class MyExecutable implements Executable {
@Override
public void execute() throws Throwable {
System.out.println("Hello World!");
}
}
```

Problem Statement-6

```
package com;
import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertTrue;
import org.junit.jupiter.api.*;
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.condition.DisabledIf;
import org.junit.jupiter.api.condition.EnabledOnOs;
import org.junit.jupiter.api.condition.OS;
import org.junit.platform.runner.~~JUnitPlatform~~;
import org.junit.runner.RunWith;
@DisplayName("JUnit 5 Dependency Injection Example")
@RunWith(~~JUnitPlatform~~.class)
public class DInjecion {
DInjecion(TestInfo testInfo) {
assertEquals("JUnit 5 Dependency Injection Example",
testInfo.getDisplayName());
}
@BeforeEach
void init(TestInfo testInfo) {
String displayName = testInfo.getDisplayName();
assertTrue(displayName.equals("TEST 1") || displayName.equals("test2()"));
}
@Test
@DisplayName("TEST 1")
@Tag("my-tag")
void test1(TestInfo testInfo) {
assertEquals("TEST 1", testInfo.getDisplayName());
assertTrue(testInfo.getTags().contains("my-tag"));
}
@Test
void test2() {
}
}
```



```
Finished after 0.188 seconds
Runs: 4/4          Errors: 0          Failures: 0
com.DInjecion [Runner: JUnit 4] (0.014 s)          Failure Trace
```